Real-Time Automatic Target Selection and Tracking Algorithm Using Keypoint-Based Template Matching

Chi-Yi Tsai^{*} and Cheng-Jui Yu

Department of Electrical Engineering, Tamkang University
No. 151, Ying-zhuan Road, Tamsui District
New Taipei City 25137, Taiwan R.O.C.

^{*}Corresponding Author: chiyi_tsai@mail.tku.edu.tw

Abstract

Object detection and tracking is an important function that helps bringing robots into real-world environments. Detection of an object-of-interest (OOI) from multiple stacking objects still holds certain degree of difficulty. In this paper, a novel target selection and tracking algorithm is proposed for detecting and tracking one OOI from multiple stacking objects. The proposed method first employs a keypoint extraction algorithm to compute control points from incoming images. Next, each OOI appeared in the image is extracted from the multiple stacking objects in a box by using mean shift clustering approach. Finally, a template-based visual tracking method is adopted to locate and track center position of the top OOI in the box. The proposed algorithm had been implemented on an Intel Core i5-4440 3.1GHz platform, achieving real-time performance about 30 frames per second at 640x480 image resolution in the experiments.

Keywords: Keypoint extraction, object detection, mean shift clustering, template matching, visual tracking.

1. Introduction

Among a variety of different approaches for robotic bin picking, computer vision has been widely used in development of vision-based object detection and tracking systems, which is an important technique to bring intelligent robots into industrial applications and daily life of human beings. Although many vision-based object detection and recognition algorithms are well studied in literature, only a few attentions have been devoted to develop detection and tracking of an object-of-interest (OOI) from multiple stacking objects in a box, which is a

fundamental operation in applications of robotic random bin picking. Therefore, this paper presents a novel target selection and tracking algorithm for multiple object separation, target selection, and target tracking. Moreover, the proposed method is computational efficiency and achieves about 30 frames per second (fps) for a video stream with size 640x480 pixels. This advantage improves the applicability of the proposed method. Experimental results validate the performance of the proposed method.

2. Problem Statement

Keypoints are image features that can be repeatedly detected in images robust to translation, scaling, and rotation transformations. To detect an OOI appeared in the image, various keypoint description approaches, such as SIFT⁽¹⁾, PCA-SIFT⁽²⁾ and SURF⁽³⁾ algorithms, have been proposed to detect the OOI in a query image. Given a reference image of the OOI, the keypoint-based object detection methods first extract a reference keypoint set from the reference image. As a new image captured from the visual sensing system, an object detection module extracts a query keypoint set of the captured image and finds the keypoint matches between the query and reference keypoint sets. To improve the object detection accuracy, a RANdom Sample Consensus (RANSAC) algorithm⁽⁴⁾ is usually applied to remove outliers in the keypoint matches. The traditional keypoint-based object detection methods are useful to detect a single OOI in a complex environment; however, they may not work well to detect multiple objects randomly placed in a box, as shown in Fig. 1. This problem motivates us to develop a real-time keypoint-based target selection and tracking system to detect the top OOI from the multiple stacking objects randomly placed in the box.

Fig. 1. The target selection problem considered in this study: multiple stacking objects are randomly placed in a box, and we have to decide which one should be taken from the box first.

Fig. 2. Flowchart of the proposed target selection and tracking system for the application of random bin picking.

3. Proposed Method

Figure 2 shows flowchart of the proposed target selection and tracking system, which consists of object detection, object separation, and object tracking steps. In the object detection step, the SURF algorithm is employed to extract keypoints in the current input image. In the object separation step, the image coordinates of the center point of each object are first estimated by a center-point voting process⁽⁵⁾. Then, the number of objects is determined by a data clustering algorithm, such as K-means⁽⁶⁾ or mean shift⁽⁷⁾ techniques. Here, we use mean shift algorithm for data clustering as it is suitable to classify multiple classes without an initial guess of the number of classes. After separating each OOI, a template-based visual tracking method⁽⁸⁾ is used to track the top OOI in real-time.

3.1 Object Detection

To detect the OOI in the image, the keypoint-based object detection method is used, which detects all possible objects based on keypoint matches between a given object reference image and an input query image. Figure 3 shows the SURF keypoints extracted from the reference and query

Fig. 3. Keypoints extracted from an object reference image (left) and an input query image (right).

images. Next, a keypoint matching set is obtained from the reference and query keypoint sets by using a fast keypoint descriptor matching algorithm⁽⁹⁾.

3.2 Object Separation

To separate the detected objects in the image, a center-point voting algorithm⁽⁵⁾ is first used to estimate the position of center point of each object. Let m_q and m_r denote the matched SURF keypoints in the query and reference images, respectively. Then, the center point c_q of a corresponding object in the query image is estimated by

$$c_q = m_q + r \cdot M_{Ori}(\phi) \cdot v_r, \quad (1)$$

where r is a scale ratio defined as

$$r = \frac{\sigma_q}{\sigma_r}, \quad (2)$$

where σ_q and σ_r denote the scale of the keypoints m_q and m_r , respectively. v_r is a reference central vector from the center point of the reference object to the matched reference keypoint m_r such that

$$v_r = m_r - c_r, \quad (3)$$

where c_r is the center point of the reference object. $M_{Ori}(\phi)$ denotes an orientation rotation matrix associated with the difference between the orientation angle of the two matched keypoints m_q and m_r such that

$$M_{Ori}(\phi) = \begin{bmatrix} \cos \phi & \sin \phi \\ -\sin \phi & \cos \phi \end{bmatrix}, \quad (4)$$

$$\phi = \theta_q - \theta_r, \quad (5)$$

where θ_q and θ_r are the orientation angle of the keypoints m_q and m_r , respectively. Figure 4 shows the center-point voting result obtained from Eq. (1). It is clear that Eq. (1)

Fig. 4. Center-point voting result obtained from Eq. (1).

Fig. 5. Estimation of the center point of each OOI by using mean shift clustering algorithm.

efficiently maps each matched query keypoint to the location near the center point of a corresponding OOI.

To estimate the number of objects and a more accurate center point of each object, the mean shift clustering algorithm is then used to separate the center-point voting result into multiple object clusters by iteratively searching the densest places of the voted center points. Figure 5 shows the object clustering result obtained from the mean shift clustering algorithm. One can see from Fig. 5 that the voted center points are separated into three groups, which is the number of objects appeared in the query image. Moreover, the red points indicate the estimated center point of each detected object, which is close to the real center point of each object.

3.3 Object Tracking

Object tracking plays an important role for the object picking task as it is able to efficiently locate the position of OOI in the image with real-time performance. Thus, after separating and locating each object in the query image, a visual tracking algorithm is used to track the top OOI, which is the object cluster having maximum voted center points. Here, we employ a template-based visual tracking algorithm because a part of the reference image around the

Fig. 6. Template patch used in the object tracking step.

center point can be used as the template patch (Fig. 6) for template matching. The performance of the proposed target selection and tracking algorithm is validated in the next section.

4. Experimental Results

The proposed algorithm was implemented in C++ with OpenCV 2.4.3 library running on a Windows 7 platform with 3.1GHz Intel Core i5-4440 processor. A box containing three randomly placed objects was used to test the performance and reliability of the proposed algorithm in the application of real-time target selection and tracking. Figure 7 presents the experimental results, in which the proposed algorithm aims to consecutively detect and track one OOI in the box. In Fig. 7(a), the three objects in the box are detected successfully, and the top object is selected as the OOI to be tracked and picked up from the box, as shown in Fig. 7(b). In Figs. 7(c) and 7(d), the second largest object cluster is then chosen as the OOI to be removed from the box. Finally, the last object is detected as the OOI to be picked up, as shown in Fig. 7(e). Therefore, these experimental results validate the performance of the proposed algorithm.

Note that, the proposed algorithm is also computationally efficient as it achieves about 30 fps when processing a standard VGA (640x480 image resolution) video stream in the experiments. This advantage improves the usability of the proposed algorithm in practical applications.

5. Conclusions

In this paper, a novel keypoint-based automatic target selection and tracking algorithm is proposed. The proposed algorithm consists of object detection, object separation, and object tracking steps. Any SIFT-like keypoint detector can be used in the object detection step of the proposed algorithm. In the object separation step, a center-point voting process combined with the mean shift clustering

Fig. 7. Experimental results of the proposed target selection and tracking algorithm to consecutively detect and track one OOI randomly placed in a box.

algorithm is proposed to estimate the number of objects and the center point position of each object. Moreover, the top object also can be found according to the number of voted center points in the classified object clusters. Finally, a

template-based visual tracking algorithm is used to track the top OOI in real-time. Experimental results validate the performance of the proposed algorithm, in terms of detection accuracy, usability, and computational efficiency.

Acknowledgment

This work was supported by the National Science Council of Taiwan, ROC under grant NSC 103-2221-E-032-068 and 103-2632-E-032-001-MY3.

References

- (1) D. G. Lowe : "Object Recognition from Local Scale-Invariant Features", IEEE International Conference on Computer Vision, pp. 1150-1157, 1999.
- (2) Y. Ke and R. Sukthakar : "PCA-SIFT: A More Distinctive Representation for Local Image Descriptors", Computer Vision and Pattern Recognition, 2004.
- (3) H. Bay, A. Ess, T. Tuytelaars and L. V. Gool : "SURF: Speeded Up Robust Features", Computer Vision and Image Understanding, Vol. 110, No. 3, pp. 346-359, 2008.
- (4) M. A. Fischler and R. C. Bolles : "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Catography", Communications of the ACM, Vol. 24, No. 6, pp. 381-395, 1981.
- (5) Z. Haladova and E. Šikudo : "Multiple Instances Detection in RGBD images", Lecture Notes in Computer Science, Vol. 8671, pp. 246-253, 2014.
- (6) J. MacQueen : "Some Methods for Classification and Analysis of Multivariate Observations", Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability 1. University of California Press, pp. 281-297, 1967.
- (7) Y. Cheng : "Mean Shift, Mode Seeking, and Clustering", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 17, No. 8, pp. 790-799, 1995.
- (8) R. Brunelli : "Template Matching Techniques in Computer Vision: Theory and Practice", ISBN 978-0-470-51706-2, 2009.
- (9) C.-Y. Tsai, A.-H. Tsao and C.-W. Wang : "Real-Time Feature Descriptor Matching via a Multi-Resolution Exhaustive Search Method", Journal of Software, Vol. 8, No. 9, pp. 2197-2201, 2013.